6.01 Midterm 1: Fall 2009

Name: Solutions	**Section:**

Enter all answers in the boxes provided.

During the exam you may:

- read any paper that you want to
- use a laptop, but **only to READ course material**

You may not

- search the web generally
- run Idle or Python
- talk, chat or email or otherwise converse with another person
- use a music player

Because of this midterm, we will not have software labs this week. Instead, we will provide lectures on control systems during your software lab time slot:

section 1	October 6	11:00am	26-100
section 2	October 6	2pm	34-101

For staff use:

1.	/10
2.	/20
3.	/20
4.	/10
5.	/20
6.	/20
total:	/100

1 OOP (10 points)

The following definitions have been entered into the Python shell:

```
class F:
    x = 0
    y = 10
    def __init__(self, y):
        self.x = y
    def a1(self, y):
        self.x = max(self.x, y)
        return self.x
    def a2(self):
        self.x = max(self.x, self.y)
        return self.x
class G(F):
    def b(self):
        self.y = self.x * self.x
        return self.y
```

Write the values of the following expressions (enter `None` when there is no value; write `Error` when an error results and explain briefly why it's an error). Assume these expressions are evaluated one after the other (in each column).

`f = F(5)`

None

`f.a1(7)`

7

`f.a1(3)`

7

`f.b()`

Error: no such method

`f.a2()`

10

`g = G()`

Error: needs argument

`g = G(3)`

None

`g.a1(5)`

5

`g.b()`

25

`g.a2()`

25

2 Signals (20 points)

Given a signal `x`, we want to construct a signal `y` such that:

```
y[n] = a*x[n-1] + (1-a)*x[n-2]
```

for $0 \leqslant a \leqslant 1$.

Part a. Define a `Signal` subclass that represents this signal, given an instance of the `Signal` class representing `x` and the value of a. Construct the new signal, given a signal instance `x`, and a variable `a`, as follows:

```
y = MySig(x, a)
```

Do not use any of the existing `Signal` subclasses, such as `Rn` or `PolyR`.

```
class MySig(sig.Signal):
  def __init__(self, x, a):
    self.x = x
    self.a = a
  def sample(self, n):
    return self.a*self.x.sample(n-1) + (1-self.a)*self.x.sample(n-2)
```

Part b. Given a signal instance `x`, and a variable `a`, write an expression involving `sig.PolyR` that constructs an equivalent signal. To construct a polynomial use `poly.Polynomial`.

```
sig.PolyR(x, poly.Polynomial([1-a, a, 0]))
```

Part c. Write a Python procedure `settleTime` that is given:

- `s`: an instance of the `Signal` class;
- `m`: an integer, representing the max index to look at;
- `bounds`: a tuple of two values `(lo, hi)`.

It returns an integer or `None`. The integer corresponds to the smallest sample index (`w`) such that the sample at index `w` and all the samples from `w` up to `m` are between `lo` and `hi` (less than or equal to `hi` and greater or equal to `lo`). It returns `None` if no such sample exists.

```
def settleTime(s, m, bounds):
    (lo,hi)=bounds
    def inBounds(x):
        return x>=lo and x<=hi
    w = 0
    for i in range(m):
        if not inBounds(s.sample(i)):
            w = i+1
    if w == m: return None
    else: return w
```

3 State Machines (20 points)

Write a state machine whose inputs are the characters of a string of "words" separated by spaces. For each input character, the machine should:

- output the string `'x'` if the character is within a word, or
- if the input character is a space, then
  - output the most recent word if the most recent word is a number (all numerical digits), or
  - output `None` if the most recent input word is empty or contains any non-digits.

Assume there is never more than one consecutive space. For example:

```
>>> x = ' v1 11 1v '
>>> FindNumbers().transduce(x)
[None, 'x', 'x', None, 'x', 'x', '11', 'x', 'x', None]
```

Do not add any instance attributes to the state machines.

If `foo` is a single-character or multi-character string, the Python method `foo.isdigit()` returns `True` if there is at least one character in `foo` and all of the characters in `foo` are all digits.

```
class FindNumbers(sm.SM):
   startState = ''
   def getNextValues(self, state, inp):
      if inp == ' ':
         if state.isdigit():
            return ('', state)
         else:
            return ('', None)
      else:
         return (state+inp, 'x')
```

4 Difference Equations (10 points)

Newton's law of cooling states that:

The change in an object's temperature from one time step to the next is proportional to the difference (on the earlier step) between the temperature of the object and the temperature of the environment, as well as to the length of the time step.

Let

- $o[n]$ be temperature of object
- $s[n]$ be temperature of environment
- T be the duration of a time step
- K be the constant of proportionality

Part a. Write a difference equation for Newton's law of cooling. Be sure the signs are such that the temperature of the object will eventually equilibrate with that of the environment.

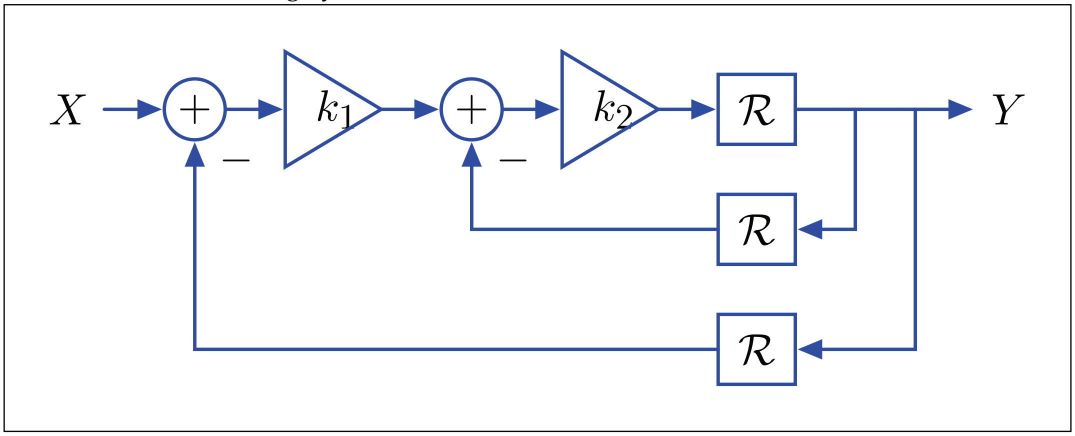
$$o[n] = o[n-1] + TK(s[n-1] - o[n-1])$$

Part b. Write the system function corresponding to this equation (show your work):

$$H = \frac{O}{S} = \frac{KTR}{1 - (1 - KT)R}$$

5 Signals and Systems (20 points)

Consider the following system:



Part a. Write the system function:

$$H = \frac{Y}{X} = \frac{k_1 k_2 R}{1 + k_2 R^2 (1 + k_1)}$$

Part b.

Let $k_1 = 1$ and $k_2 = -2$. Assume that the system starts "at rest" (all signals are zero) and that the input signal X is the unit sample signal. Determine $y[0]$ through $y[3]$.

$y[0] =$ **0**

$y[1] =$ **-2**

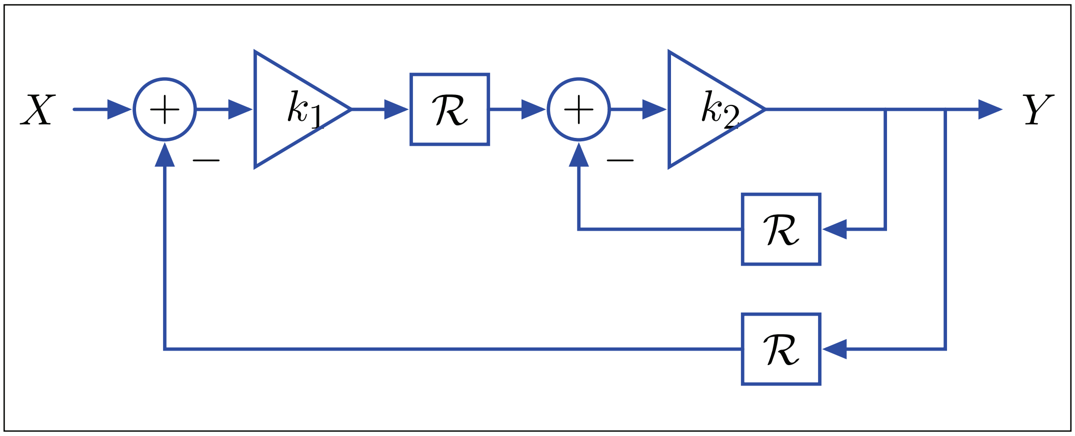
$y[2] =$ **0**

$y[3] =$ **-8**

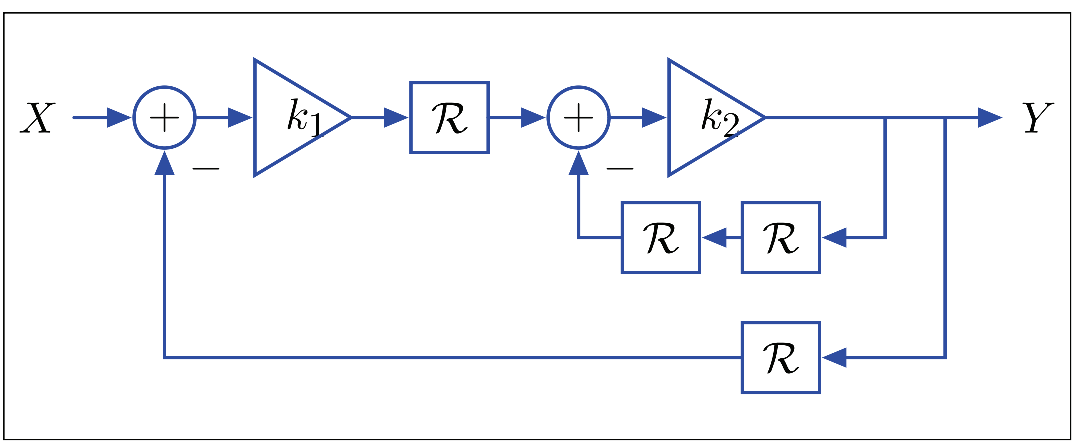
Part c. Let $k_1 = 1$ and $k_2 = -2$, determine the poles of H.

Enter poles or **none** if there are no poles: 2, -2

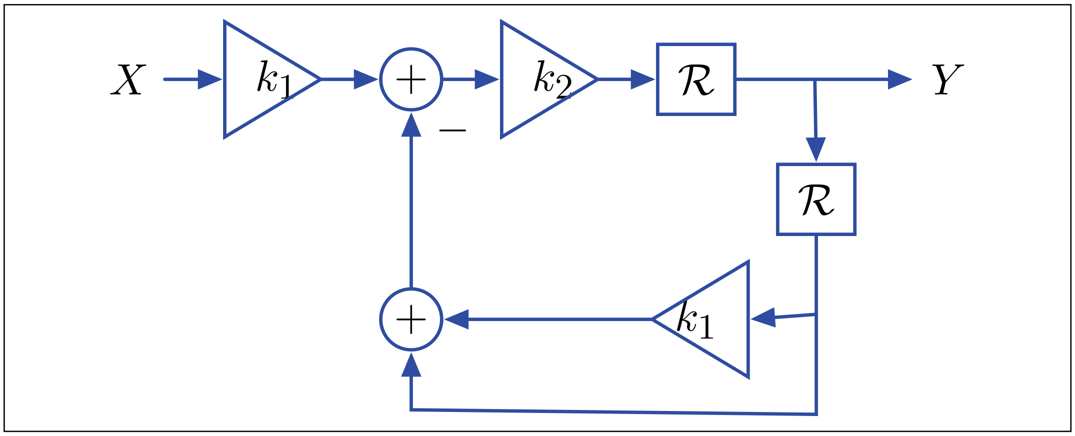
Part d. For each of the systems below indicate whether the system is equivalent to this one: (Remember that there can be multiple "equivalent" representations for a system.) If you write **clearly** the system function for these systems, we may be able to give you partial credit.



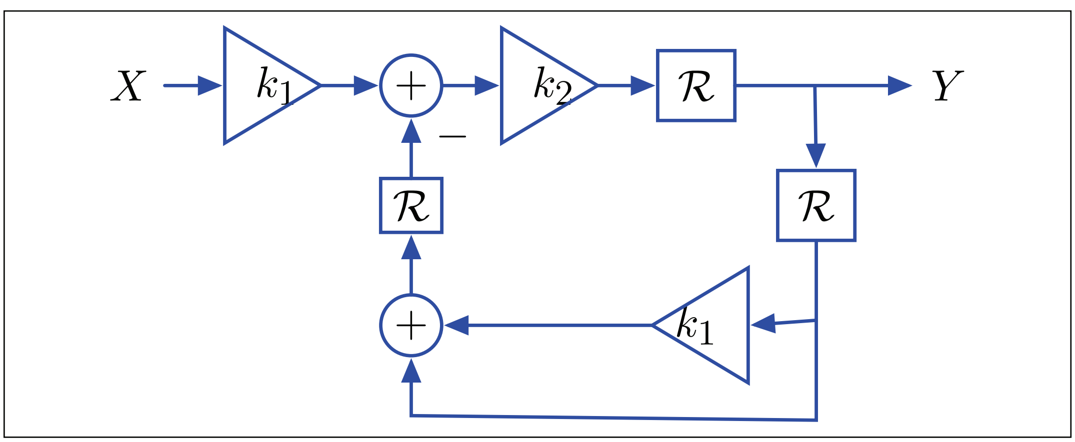
Equivalent to H (yes/no)? **No**



Equivalent to H (yes/no)? **Yes**



Equivalent to H (yes/no)? **Yes**



Equivalent to H (yes/no)? **No**

6 System Behaviors (20 points)

Part a. Find the poles for the following system functions:

$$H_1(\mathcal{R}) = \frac{1}{1 - \mathcal{R} + 0.5\mathcal{R}^2}$$

Poles: $0.5 \pm 0.5j$

$$H_2(\mathcal{R}) = \frac{1}{1 - 0.4\mathcal{R} - 0.05\mathcal{R}^2}$$

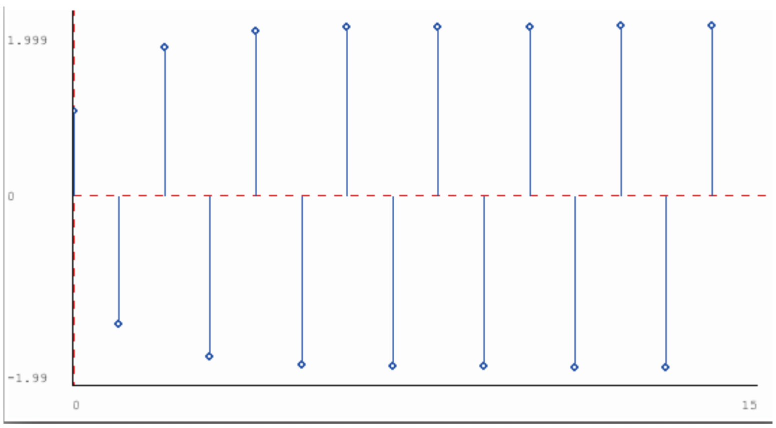
Poles: $0.5, -0.1$

Consider the following poles:

```
>>> H1.poles() = [-1.0, -0.5]
>>> H2.poles() = [(0.375+0.330j), (0.375-0.330j)]
>>> H3.poles() = [(0.05+0.234j), (0.05-0.234j)]
>>> H4.poles() = [1.2, -0.5]
```

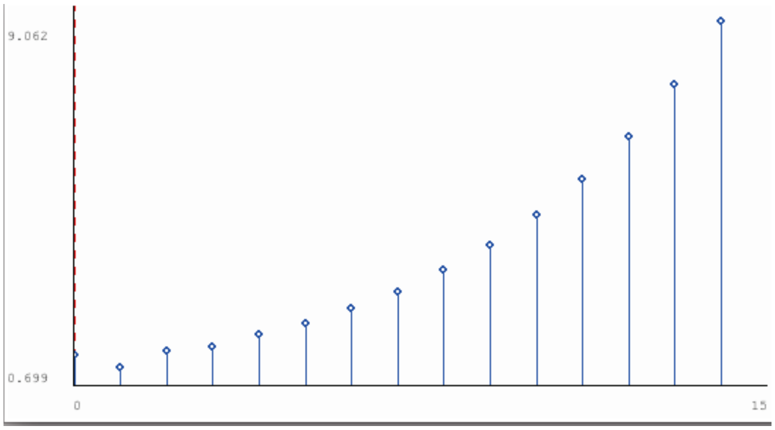
Part b. Deleted

Part c. Which (if any) of the poles lead to the following unit sample response?



Circle all correct answers(s): H_1 H_2 H_3 H_4 **none**

Part d. Which (if any) of the poles lead to the following unit sample response?



Circle all correct answers(s): H_1 H_2 H_3 H_4 **none**

Part e. Which (if any) of the poles lead to convergent unit-sample responses?

Circle all correct answers(s): H_1 H_2 H_3 H_4 **none**